

## Tutorial Semana Oministack 10.0

### Rockseat

Produto: Aplicativo mobile usando Javascript

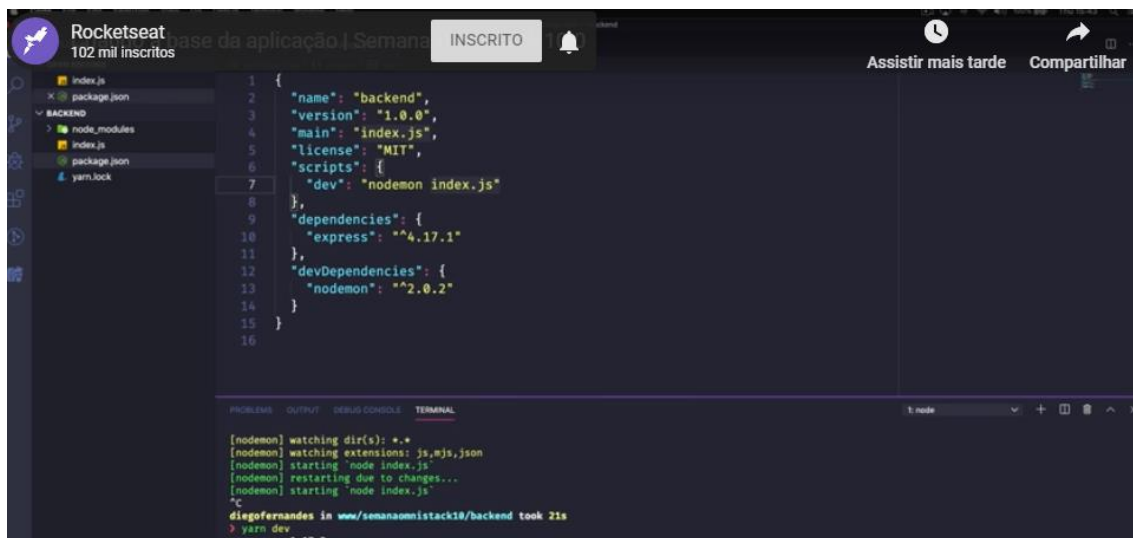
Dev: Barbara Rodrigues

Rotas get



```
1 const express = require('express');
2
3 const app = express();
4
5 // |
6
7 app.get('/', (request, response) => {
8   return response.json({ message: 'Hello OmniStack' });
9 });
10
11 app.listen(3333);
12
```

Biblioteca nodemon



Rocketseat  
102 mil inscritos

base da aplicação | Semana

INSCRITO

Assistir mais tarde Compartilhar

```
1 {
2   "name": "backend",
3   "version": "1.0.0",
4   "main": "index.js",
5   "license": "MIT",
6   "scripts": {
7     "dev": "nodemon index.js"
8   },
9   "dependencies": {
10    "express": "^4.17.1"
11  },
12  "devDependencies": {
13    "nodemon": "^2.0.2"
14  }
15 }
16
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1. node

```
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting 'node index.js'
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
^C
diegofernandes in www/semanaomnistack18/backend took 21s
$ yarn dev
yarn run v1.22.1
```

04/01/2020

## Métodos http

```
const express = require('express');

const app = express();

app.use(express.json());

// Métodos HTTP: GET, POST, PUT, DELETE

// Tipos de parâmetros:

// Query Params: request.query (Filtros, ordenação, paginação, ...)
// Route Params: request.params (Identificar um recurso na alteração ou remoção)
// Body: request.body (Dados para criação ou alteração de um registro)

app.post('/users', (request, response) => {
  console.log(request.body);
  return response.json({ message: 'Hello OmniStack' });
});

app.listen(3333);
```

## Query params

Criando a base da aplicação | Semana OmniStack 10.0

Assistir mais tarde Compartilhar

```
5 // Métodos HTTP: GET, POST, PUT, DELETE
6
7 // Tipos de parâmetros:
8
9 // Query Params: request.query (Filtros, ordenação, paginação, ...)
10 // Route Params:
11 // Body:
12
13 app.get('/users', (request, response) => {
14   console.log(request.query);
15   return response.json({ message: 'Hello OmniStack' });
16 });
17
18 app.listen(3333);
19
```

MAIS VÍDEOS

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node, rah

```
yarn run v1.17.3
$ nodemon index.js
[nodemon] 2.0.2
[nodemon] to restart at any time, enter 'rs'
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting 'node index.js'
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
```

diegofernandes in www.semanaomnistack10/backend  
>

## Instalar mongodb

Criando a base da aplicação | Semana OmniStack 10.0

```
1 const express = require('express');
2 const mongoose = require('mongoose');
3
4 const app = express();
5
6 mongoose.connect('mongodb+srv://omnistack:omnistack@cluster0-kdxe.mongodb.net/week10?retryWrites=true&w=majority', {
7   useNewUrlParser: true,
8   useUnifiedTopology: true,
9 });
10
11 app.use(express.json());
12
13 // Métodos HTTP: GET, POST, PUT, DELETE
14
15 // Tipos de parâmetros:
16
17 // Query Params: request.query (Filtros, ordenação, paginação, ...)
18 // Route Params: request.params (Identificar um recurso na alteração ou remoção)
19 // Body: request.body (Dados para criação ou alteração de um registro)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

[nodemon] restarting due to changes...  
[nodemon] starting 'node index.js'  
{ name: 'Diego', email: 'diego@rocketseat.com.br' }  
[nodemon] restarting due to changes...  
[nodemon] starting 'node index.js'  
[nodemon] restarting due to changes...  
[nodemon] starting 'node index.js'  
(node:74364) DeprecationWarning: current URL string parser is deprecated, and will be removed in a future version. To use the new parser, pass option { useNewUrlParser: true } to mongoose.connect.  
[nodemon] restarting due to changes...  
[nodemon] starting 'node index.js'

MAIS VÍDEOS

DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discovery and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.  
[nodemon] restarting due to changes...  
[nodemon] starting 'node index.js'

46:22 / 1:32:48

karere@2.3.1  
- memory-pager@1.5.0  
- mongodb@3.5.1  
- mongoose-legacy-pluralize@1.0.2  
- mongoose@5.8.6  
- npath@0.6.4  
- query@0.2.2  
- regexp-clone@1.0.0  
- require\_optional@1.0.1  
- resolve-from@2.0.0  
- sass@1.0.3  
- sift@7.0.1  
- sparse-bitfield@3.0.3  
+ Done in 0.57s.

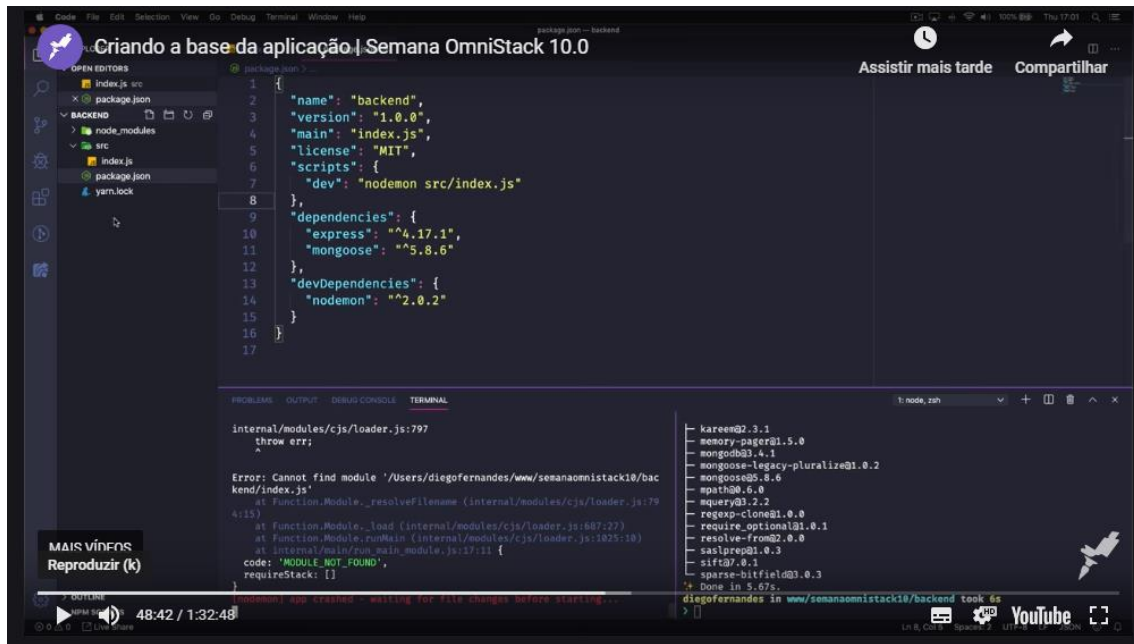
diegofernanandes in www.semanaomnistack10/backend took 6s

Verificar proxy

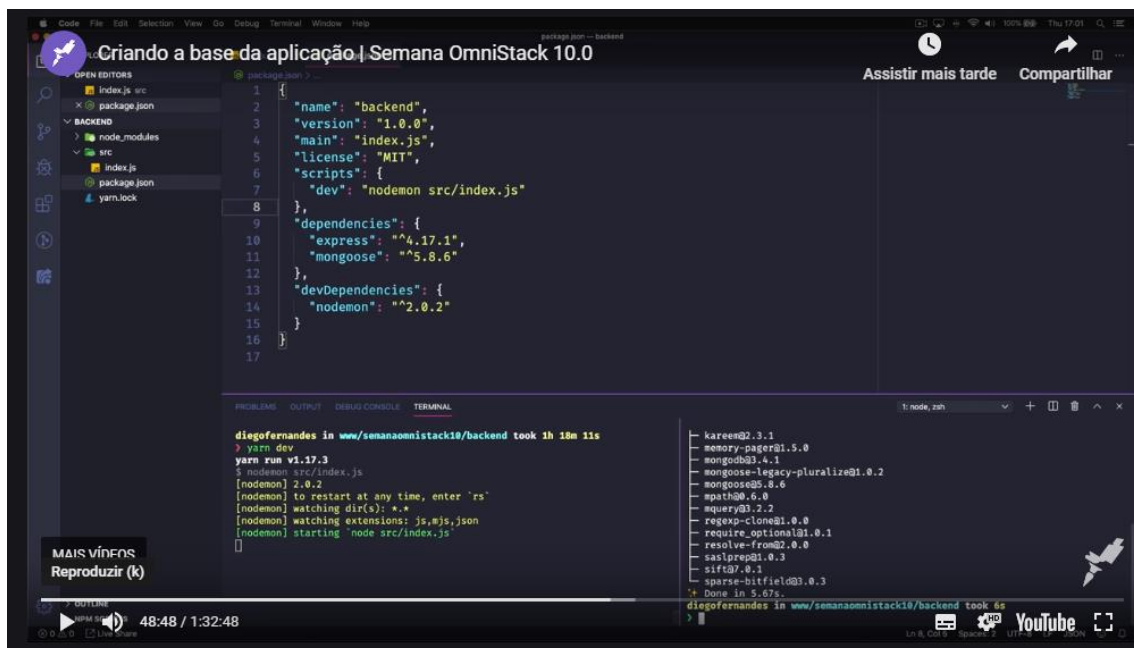


Criar pasta src com index e modificar código com src

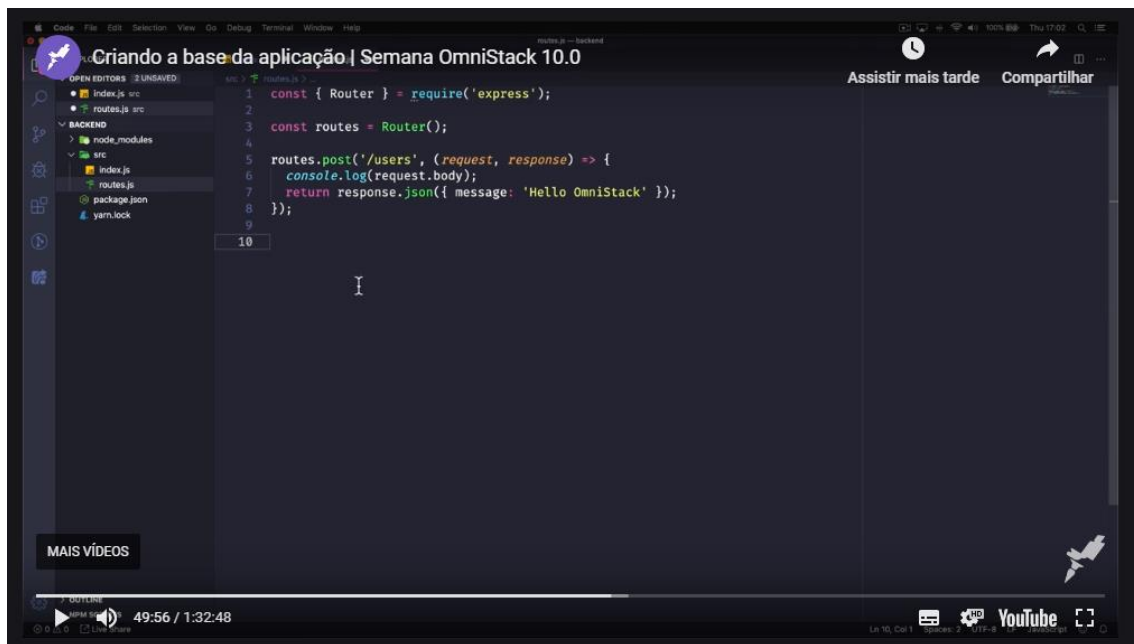
04/01/2020



Executa yarn dev



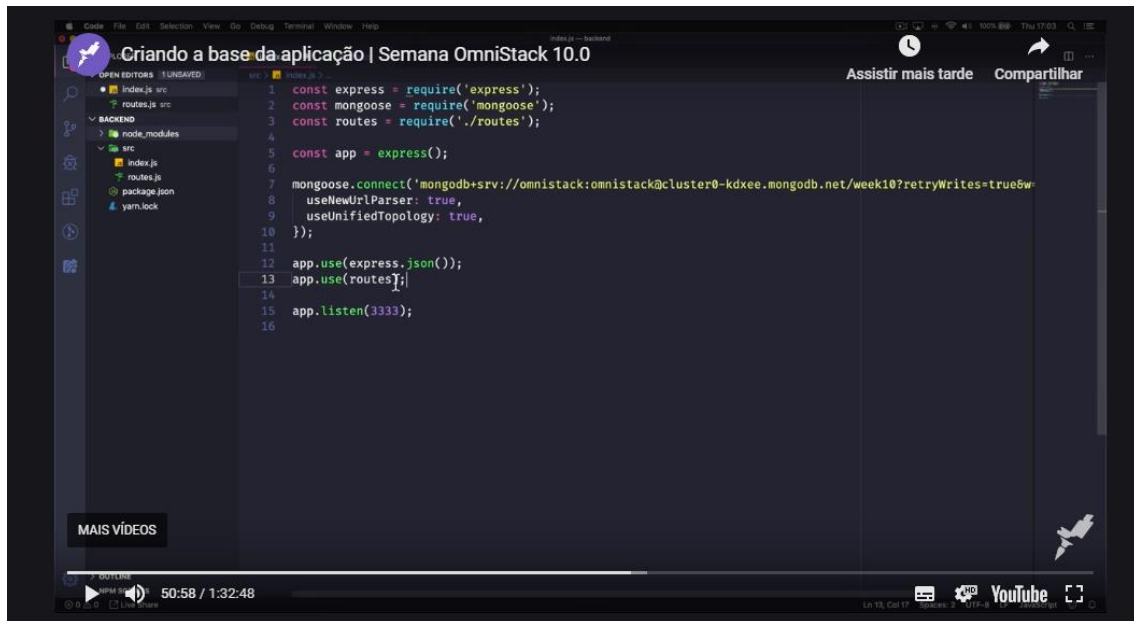
Separar rotas



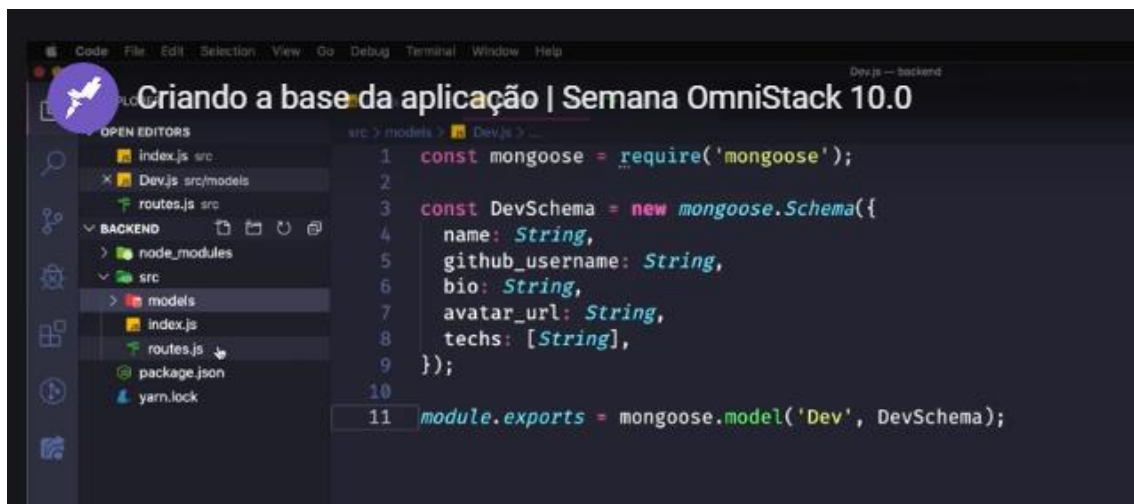
Rotas e exportar / trocar app por routes. Copiar código (get...)



Rotas alterar o index para rodar



Models



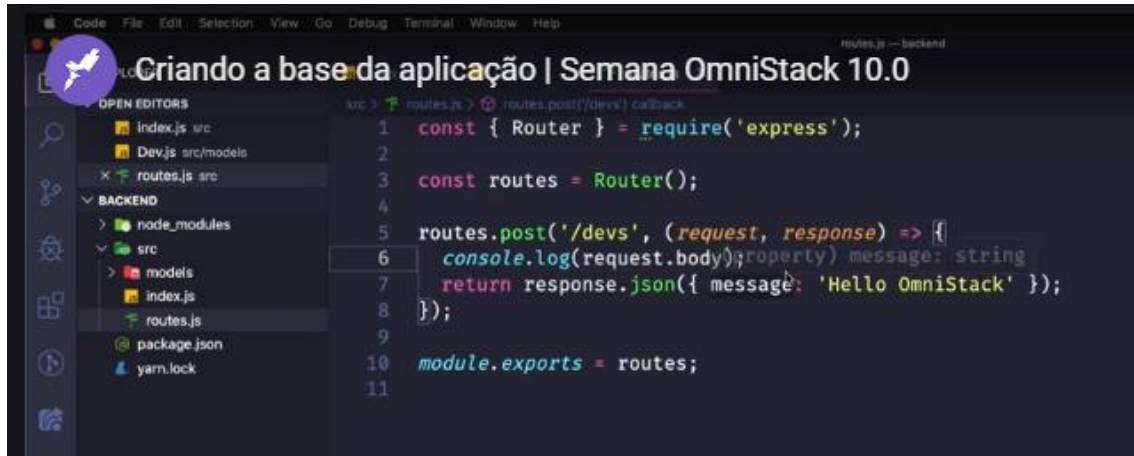
Teste insomnia



04/01/2020



Rota para buscar devs e ver insomnia



```
1 const { Router } = require('express');
2
3 const routes = Router();
4
5 routes.post('/devs', (request, response) => {
6   console.log(request.body);
7   return response.json({ message: 'Hello OmniStack' });
8 });
9
10 module.exports = routes;
```

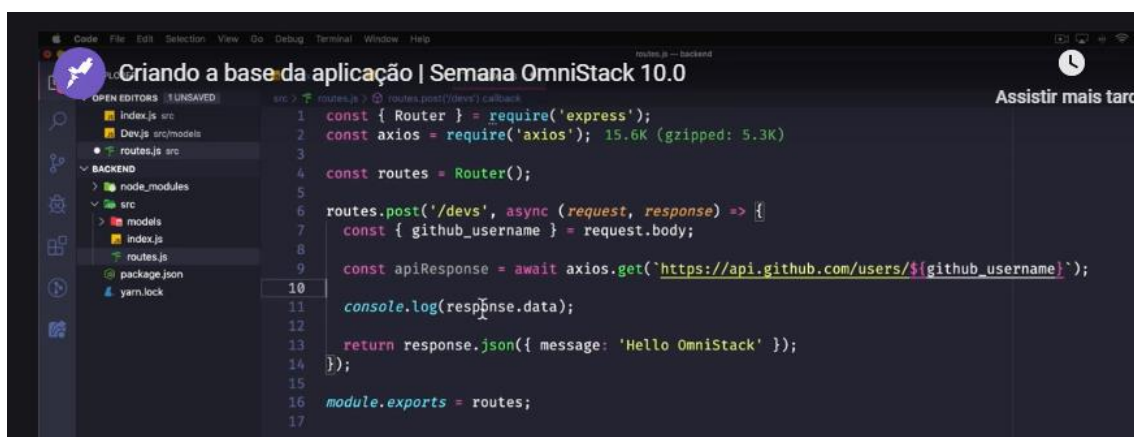
Instalar biblioteca axios para buscar dados no github



```
1 const { Router } = require('express');
2
3 const routes = Router();
4
5 routes.post('/devs', (request, response) => {
6   const { github_username } = request.body;
7
8   // ... (axios code) ...
9
10   return response.json({ message: 'Hello OmniStack' });
11 });
12
13 module.exports = routes;
```

```
diegofernandes in www/semnaomistack10/backend took 6s
> yarn add axios
```

Axios e github

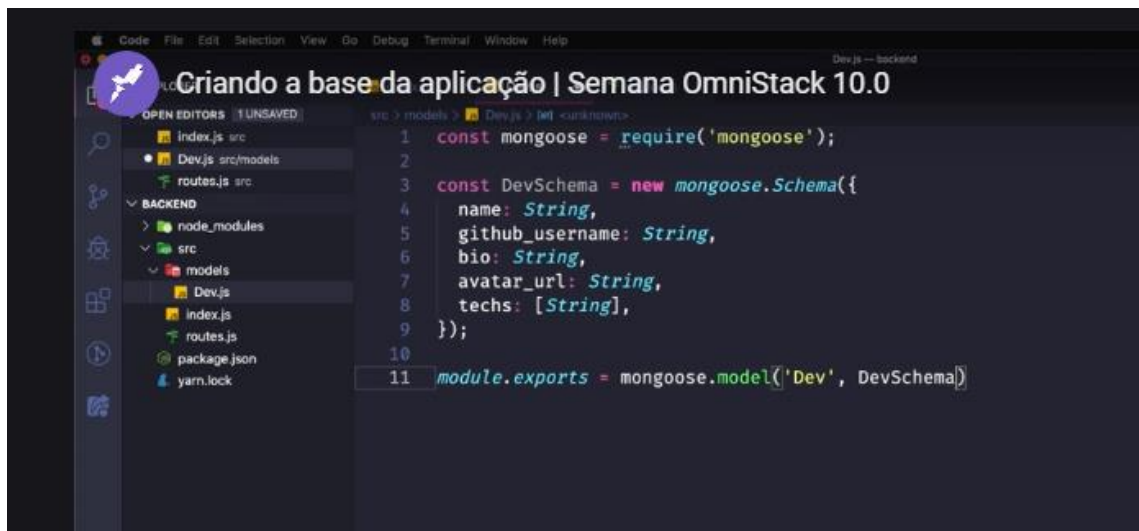


```
1 const { Router } = require('express');
2 const axios = require('axios');
3
4 const routes = Router();
5
6 routes.post('/devs', async (request, response) => {
7   const { github_username } = request.body;
8
9   const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);
10   console.log(apiResponse.data);
11   return response.json({ message: 'Hello OmniStack' });
12 });
13
14 module.exports = routes;
```

Insomnia p testar



Schema – dev.js




Cadastrar devs



04/01/2020

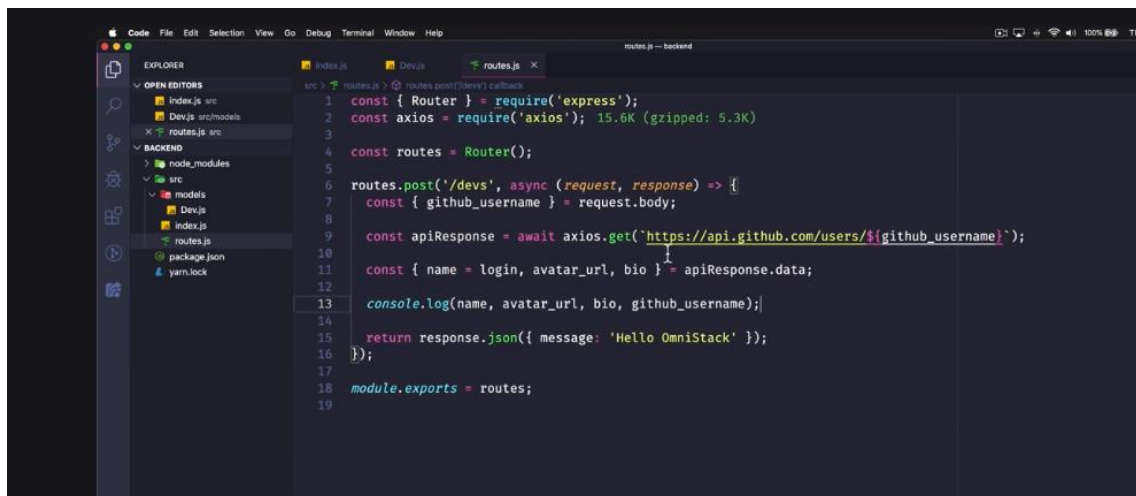


## Criando as Techs



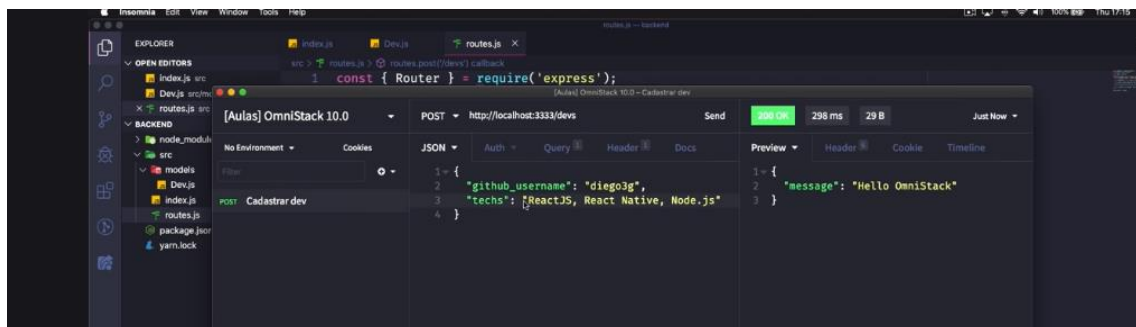
```
1 const { Router } = require('express');
2 const axios = require('axios');
3 const Dev = require('./models/Dev');
4
5 const routes = Router();
6
7 routes.post('/devs', async (request, response) => {
8   const { github_username, techs } = request.body;
9
10  const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);
11
12  const { name = login, avatar_url, bio } = apiResponse.data;
13
14  const techsArray = techs.split(',').map(tech => tech.trim());
15
16  const dev = await Dev.create({
17    github_username,
18    name,
19    avatar_url,
20    bio,
21    techs: techsArray,
22  });
23
24  return response.json(dev);
25 });
26
27 module.exports = routes;
```

## Para testar console.log



```
1 const { Router } = require('express');
2 const axios = require('axios');
3
4 const routes = Router();
5
6 routes.post('/devs', async (request, response) => {
7   const { github_username } = request.body;
8
9   const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);
10
11   const { name = login, avatar_url, bio } = apiResponse.data;
12
13   console.log(name, avatar_url, bio, github_username);
14
15   return response.json({ message: 'Hello OmniStack' });
16 });
17
18 module.exports = routes;
```

## Enviar techs por strings



Method	URL	Status	Time	Size	When
POST	http://localhost:3333/devs	200 OK	298 ms	29 B	Just Now

Header	Value
Content-Type	application/json

Body
{ "github_username": "diego3g", "techs": "ReactJS, React Native, Node.js" }

Header	Value
Content-Type	application/json

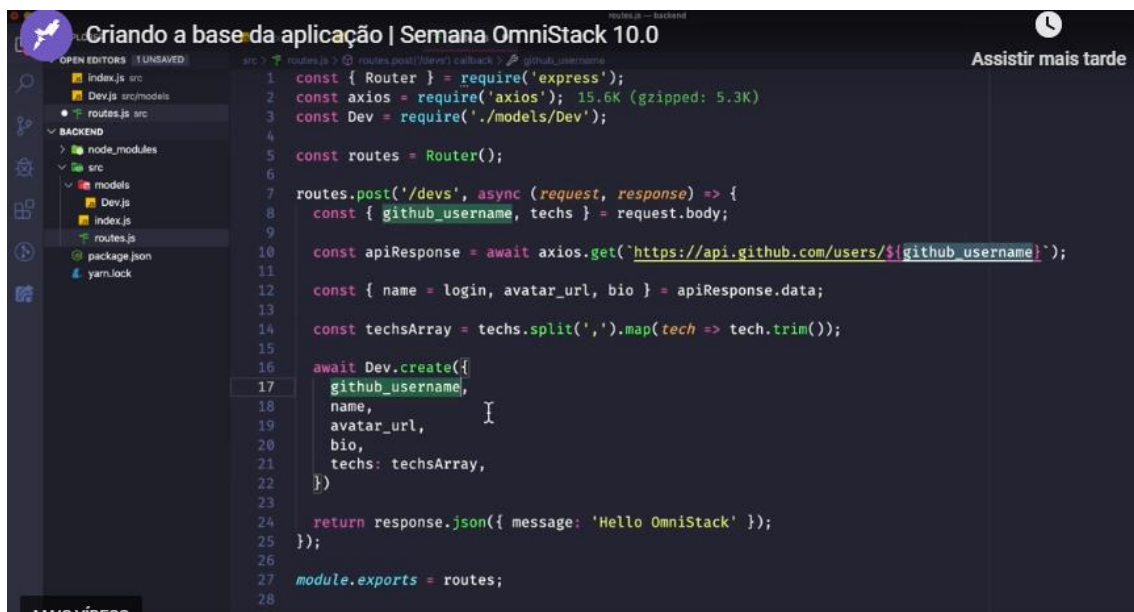
Body
{ "message": "Hello OmniStack" }

Vetor de techs para converter. Cortar string qd tiver virgula – tech.trim (remove espaçamento antes e depois)



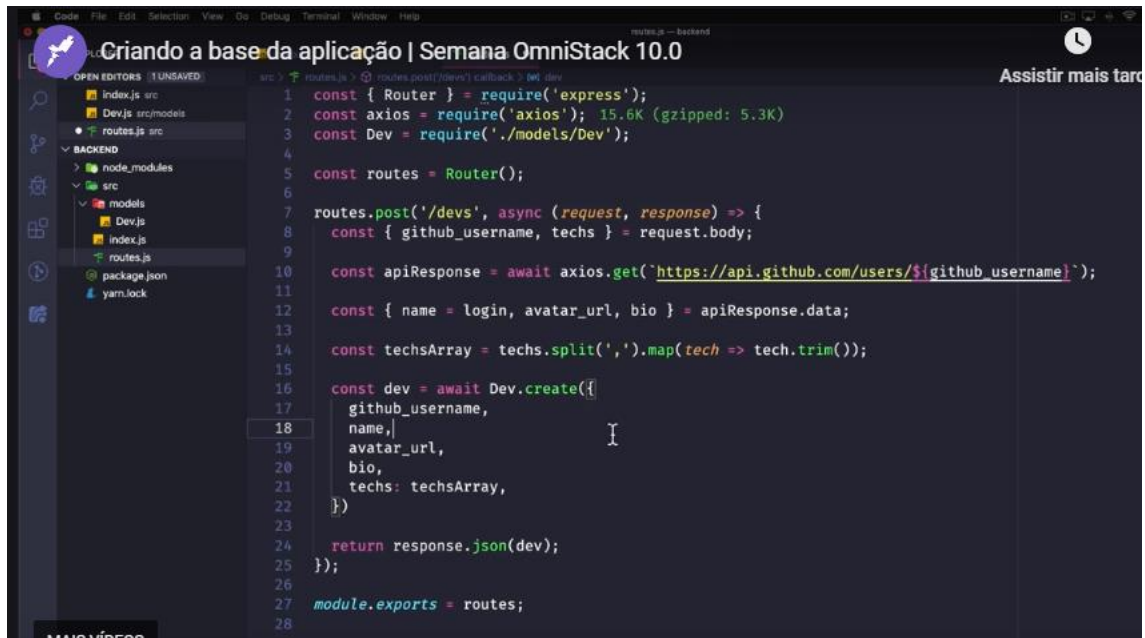
```
1 const { Router } = require('express');
2 const axios = require('axios'); // 15.6K (gzipped: 5.3K)
3
4 const routes = Router();
5
6 routes.post('/devs', async (request, response) => {
7   const { github_username, techs } = request.body;
8
9   const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);
10
11   const { name = login, avatar_url, bio } = apiResponse.data;
12
13   const techsArray = techs.split(',').map(tech => tech.trim());
14
15   return response.json({ message: 'Hello OmniStack' });
16 });
17
18 module.exports = routes;
```

Short sintaxe – não precisa usar : pq a propriedade é igual o nome da variável



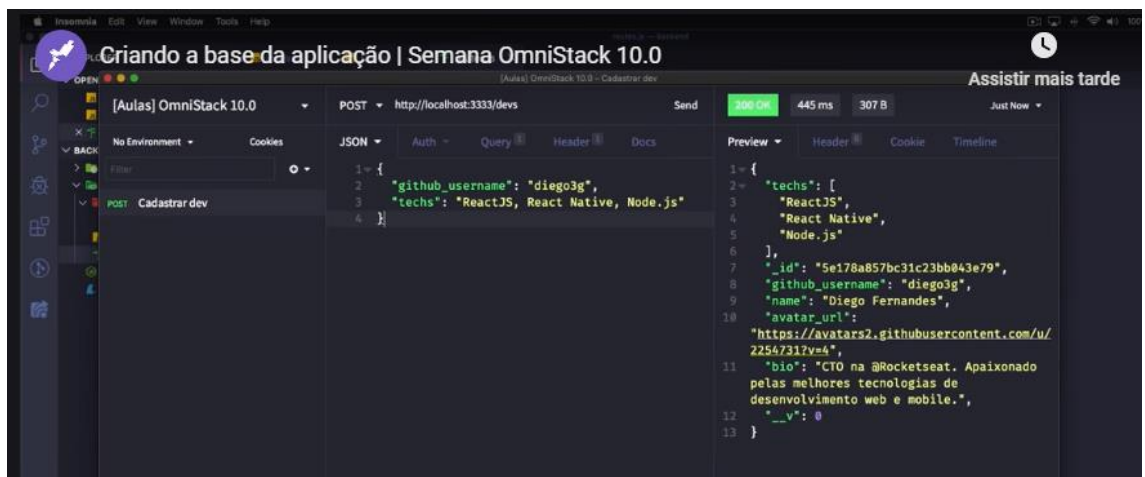
```
1 const { Router } = require('express');
2 const axios = require('axios'); // 15.6K (gzipped: 5.3K)
3 const Dev = require('./models/Dev');
4
5 const routes = Router();
6
7 routes.post('/devs', async (request, response) => {
8   const { github_username, techs } = request.body;
9
10   const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);
11
12   const { name = login, avatar_url, bio } = apiResponse.data;
13
14   const techsArray = techs.split(',').map(tech => tech.trim());
15
16   await Dev.create({
17     github_username,
18     name,
19     avatar_url,
20     bio,
21     techs: techsArray,
22   });
23
24   return response.json({ message: 'Hello OmniStack' });
25 });
26
27 module.exports = routes;
```

Código pronto p rodar e testar

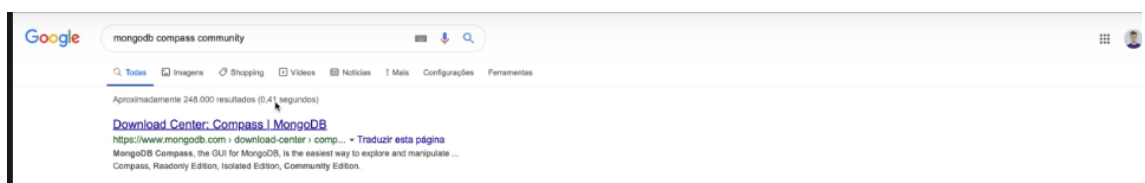


```
1 const { Router } = require('express');
2 const axios = require('axios');
3 const Dev = require('./models/Dev');
4
5 const routes = Router();
6
7 routes.post('/devs', async (request, response) => {
8   const { github_username, techs } = request.body;
9
10  const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);
11
12  const { name, login, avatar_url, bio } = apiResponse.data;
13
14  const techsArray = techs.split(',').map(tech => tech.trim());
15
16  const dev = await Dev.create({
17    github_username,
18    name,
19    avatar_url,
20    bio,
21    techs: techsArray,
22  });
23
24  return response.json(dev);
25 });
26
27 module.exports = routes;
```

Cadastro do dev no banco de dados testado no insomnia. Gerou um id – identificador único e versionamento qd tem alterações. Sem alteração 0 no mongodb

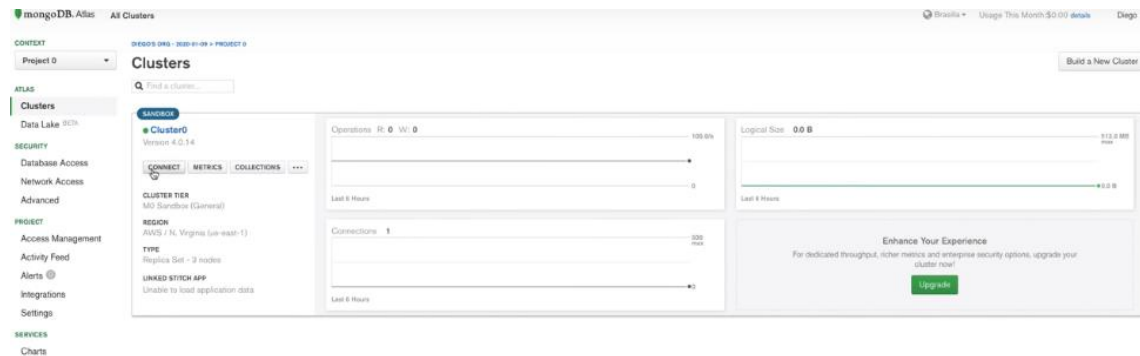


Cliente p acessar mongodb

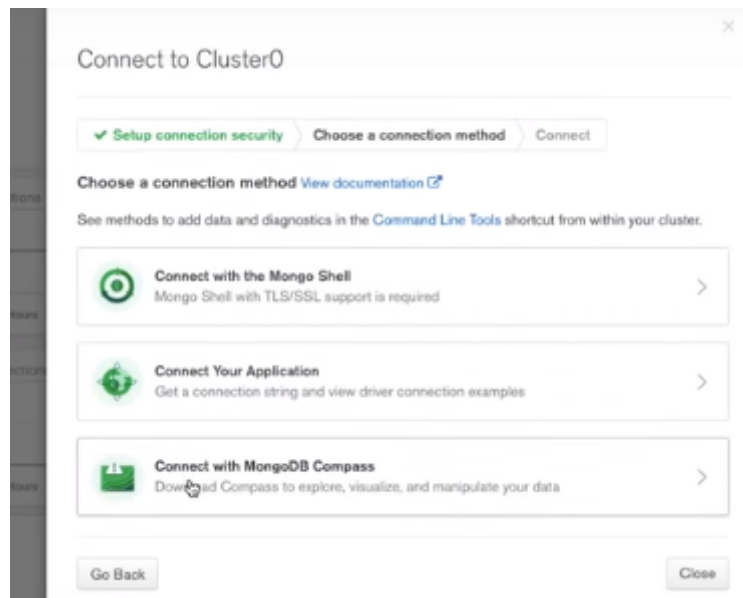


04/01/2020

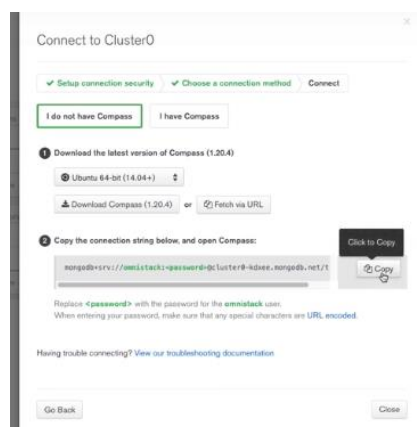
## Conexao com cluster



Clicar na opção com a mão




Copia o link



Comnodb compress identificado

04/01/2020

Connect to Host



**MongoDB connection string detected**  
 Compass detected a MongoDB connection string in your clipboard. Do you want to use the connection string to fill out this form?

☐ No ☒ Yes

Port: 27017

SRV Record: ☐

Authentication: None

Replica Set Name:

Read Preference: Primary

SSL: None

SSH Tunnel: None

Favorite Name: e.g. Shared Dev, QA Box, PRODUCTION

**CONNECT**

## Preencher campos

Hostname: cluster0-kidex.mongodb.net

SRV Record: ☒

Authentication: Username / Password

Username: omnistack

Password:

Authentication Database: admin

Replica Set Name:

Read Preference: Primary

SSL: System CA / Atlas Deployment

SSH Tunnel: None

Favorite Name: e.g. Shared Dev, QA Box, PRODUCTION




**CONNECT**

## Base de dados conectada

Cluster: Shared-0 / REPLICASET (3 NODES) mongodb 4.4.11 - omnistack

**Databases**

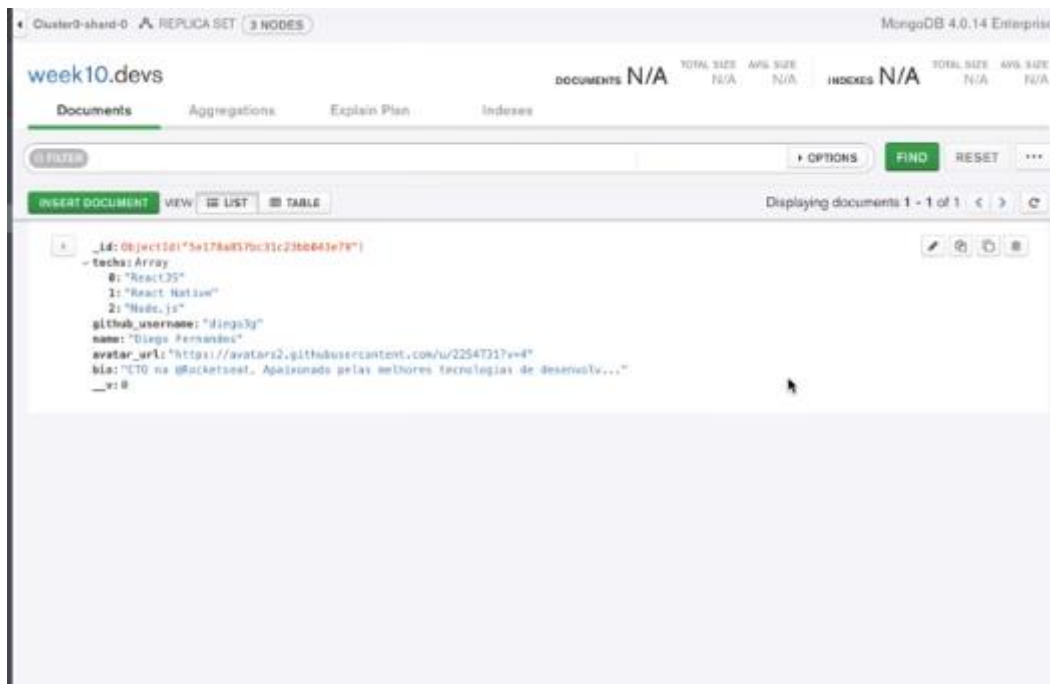
[CREATE DATABASE](#)

Database Name	Storage Size	Collections	Indexes	
admin	0.0B	0	0	
local	0.0B	6	0	
week10	16.0KB	1	1	

04/01/2020

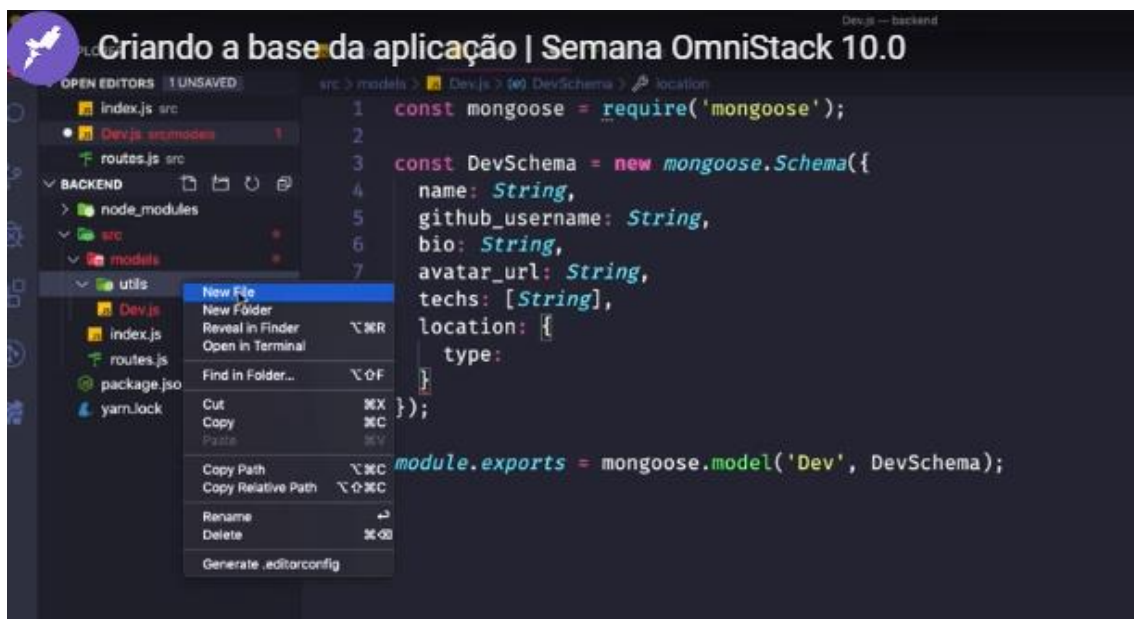


Coleções = tabelas – dev. Salvas no banco de dados.

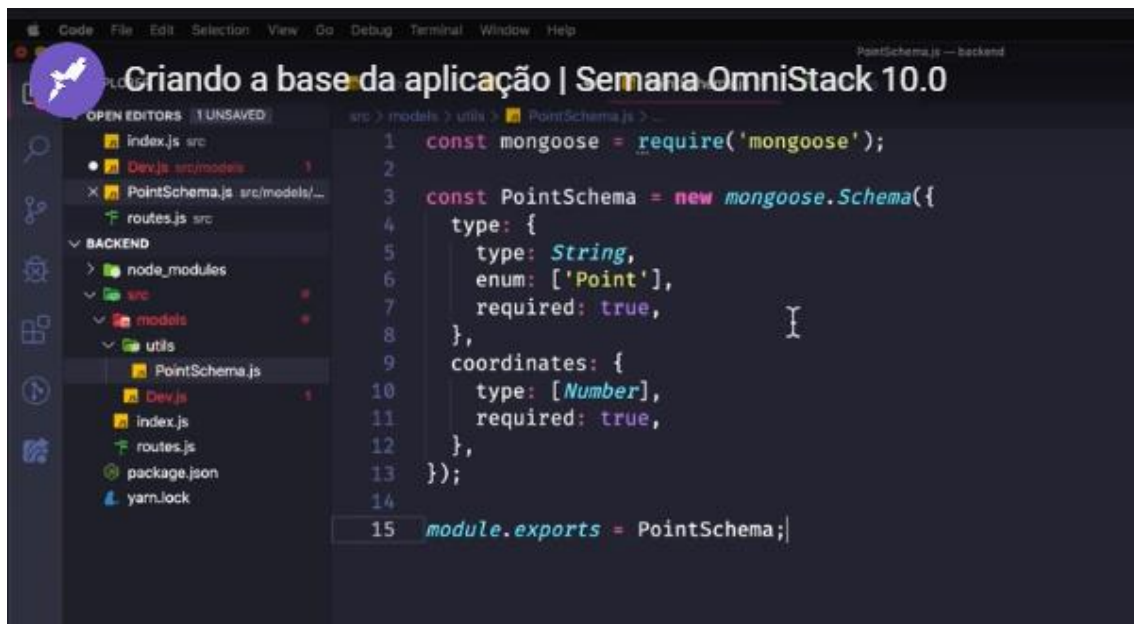


Latitude e longitude

Criar pasta e dev.js

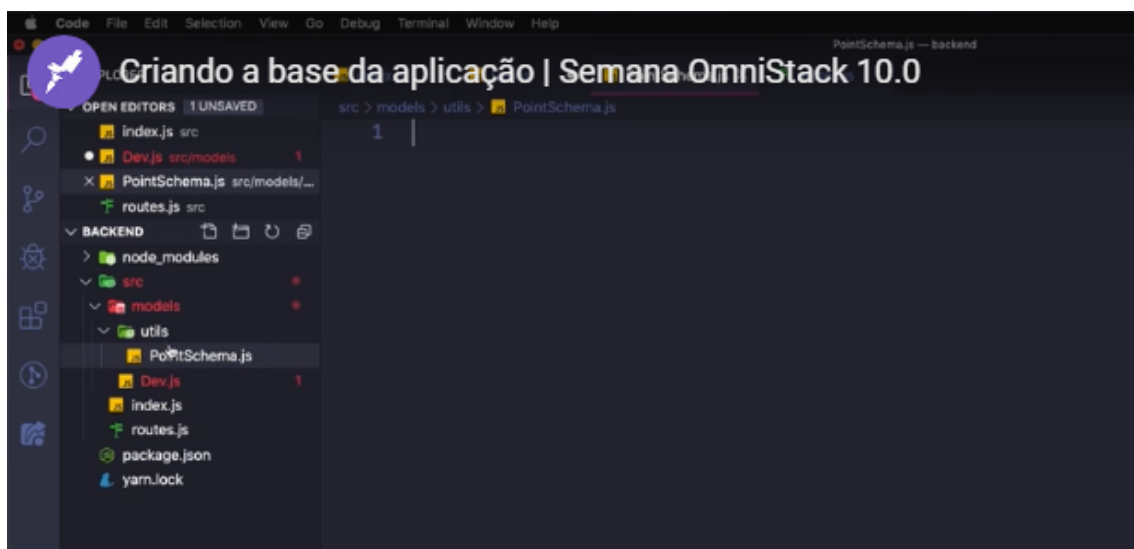


PointSchema.js para usar com utilidade



Criando a base da aplicação | SemanaOmniStack 10.0

```
1 const mongoose = require('mongoose');
2
3 const PointSchema = new mongoose.Schema({
4   type: {
5     type: String,
6     enum: ['Point'],
7     required: true,
8   },
9   coordinates: {
10    type: [Number],
11    required: true,
12  },
13 });
14
15 module.exports = PointSchema;
```



Criando a base da aplicação | SemanaOmniStack 10.0

```
1
```

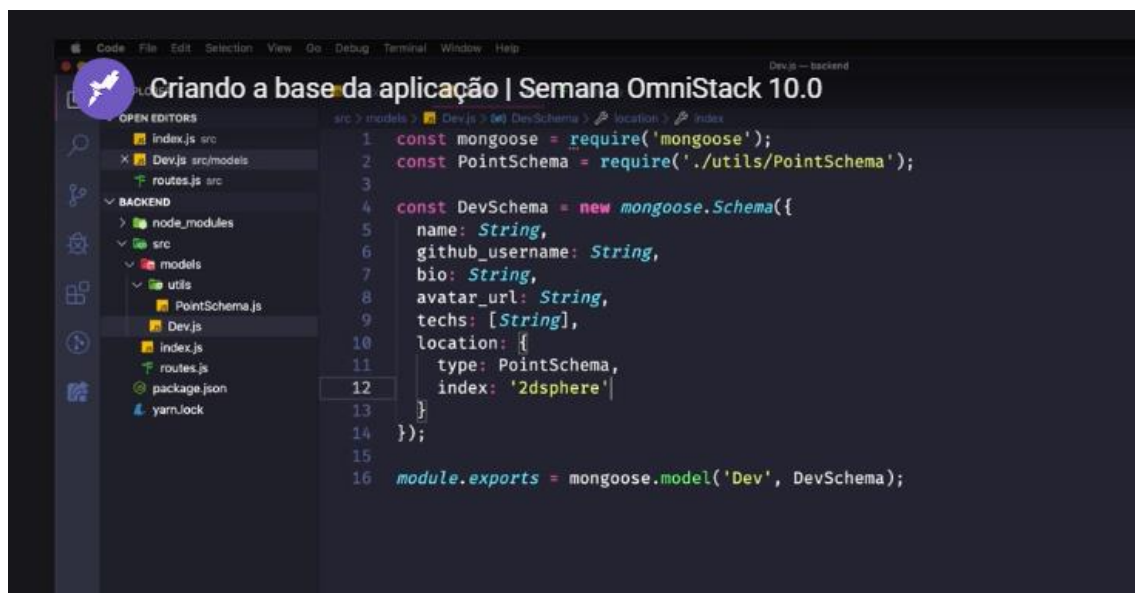
Coordinates type longitude e latitude – tem na documentação do mgdb



Criando a base da aplicação | Semana OmniStack 10.0

```
1 const mongoose = require('mongoose');
2
3 const PointSchema = new mongoose.Schema({
4   type: {
5     type: String,
6     enum: ['Point'],
7     required: true,
8   },
9   coordinates: {
10    type: [Number],
11    required: true,
12  },
13 });
14
15 module.exports = PointSchema;
```

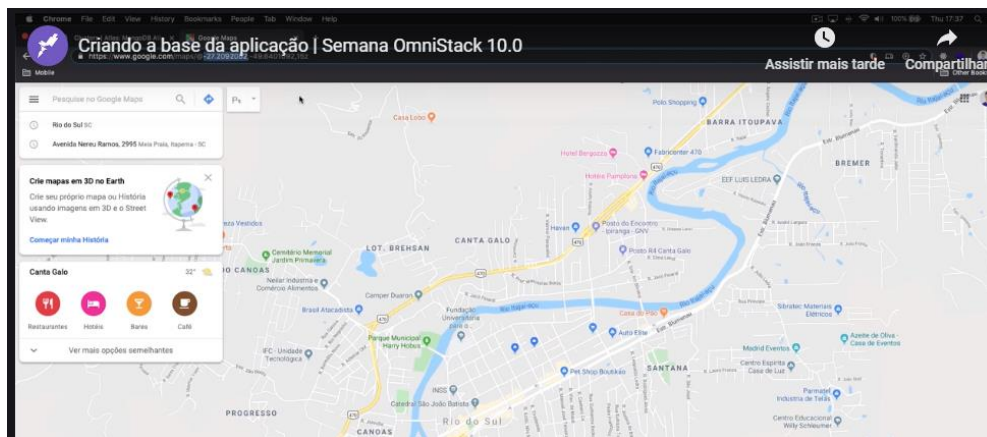
Importar e usar index



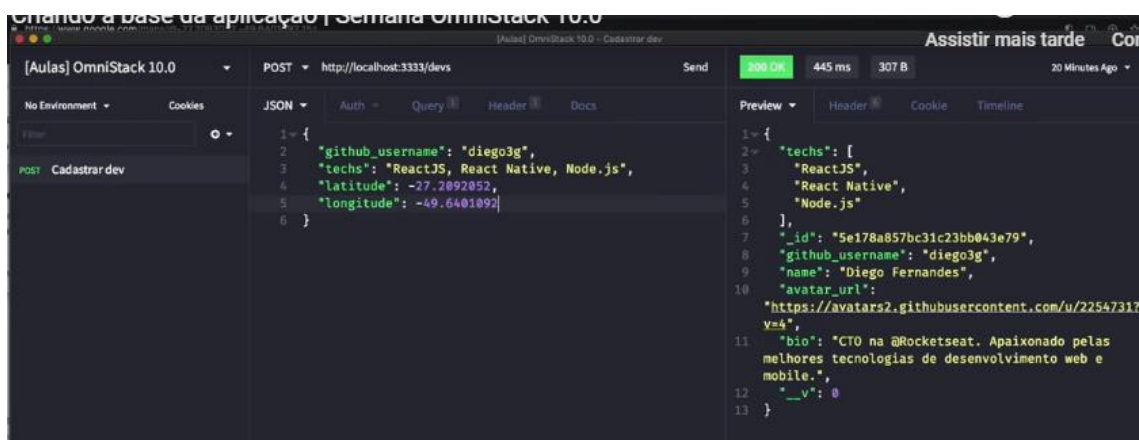
Criando a base da aplicação | Semana OmniStack 10.0

```
1 const mongoose = require('mongoose');
2 const PointSchema = require('./utils/PointSchema');
3
4 const DevSchema = new mongoose.Schema({
5   name: String,
6   github_username: String,
7   bio: String,
8   avatar_url: String,
9   techs: [String],
10  location: {
11    type: PointSchema,
12    index: '2dsphere'
13  },
14 });
15
16 module.exports = mongoose.model('Dev', DevSchema);
```

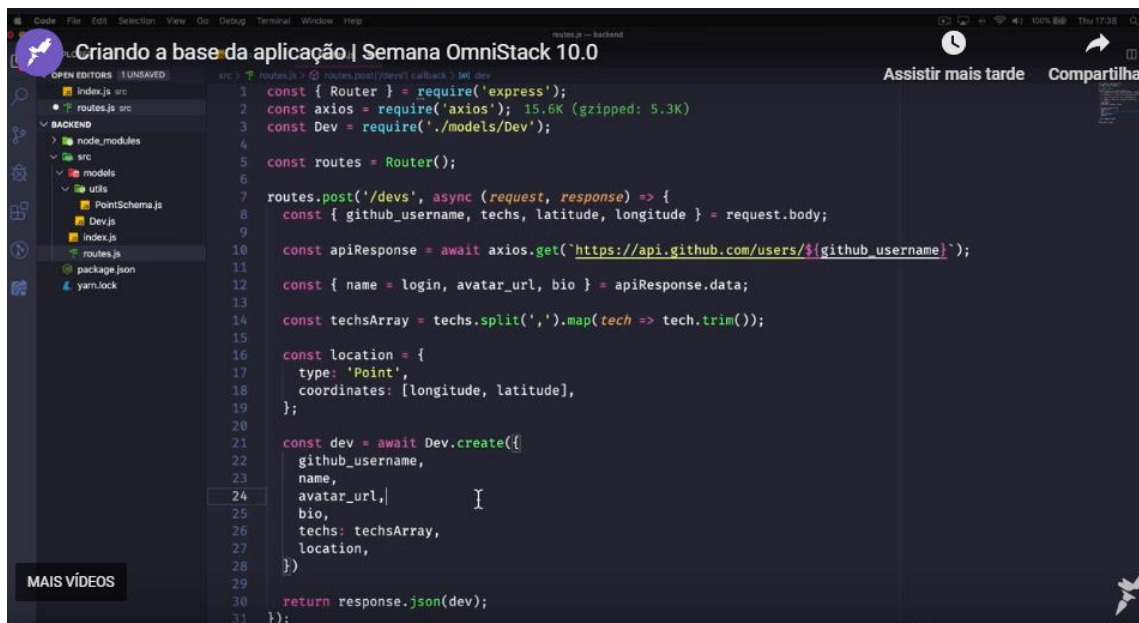
Map primeiro numero latitude e segundo longitude



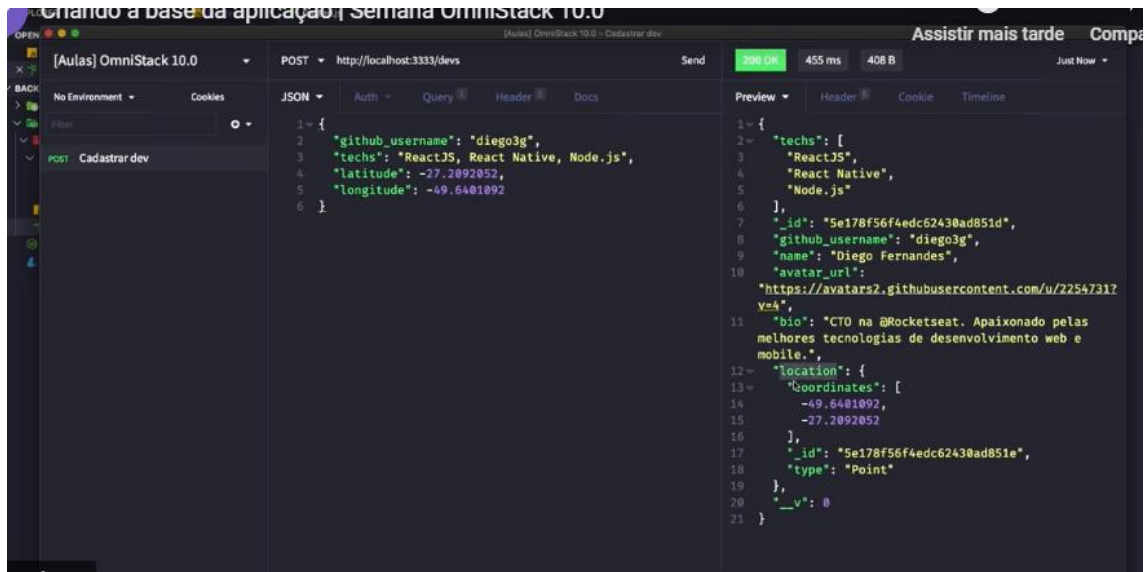
teste



Alterar esse documento com dados de localition mesmo nome no banco de dados

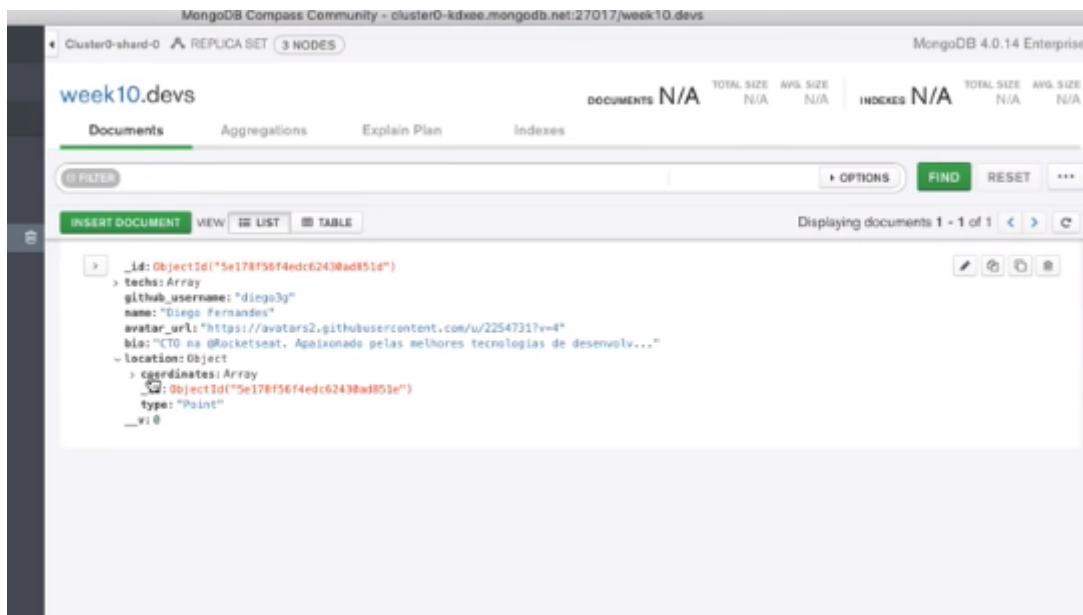


teste



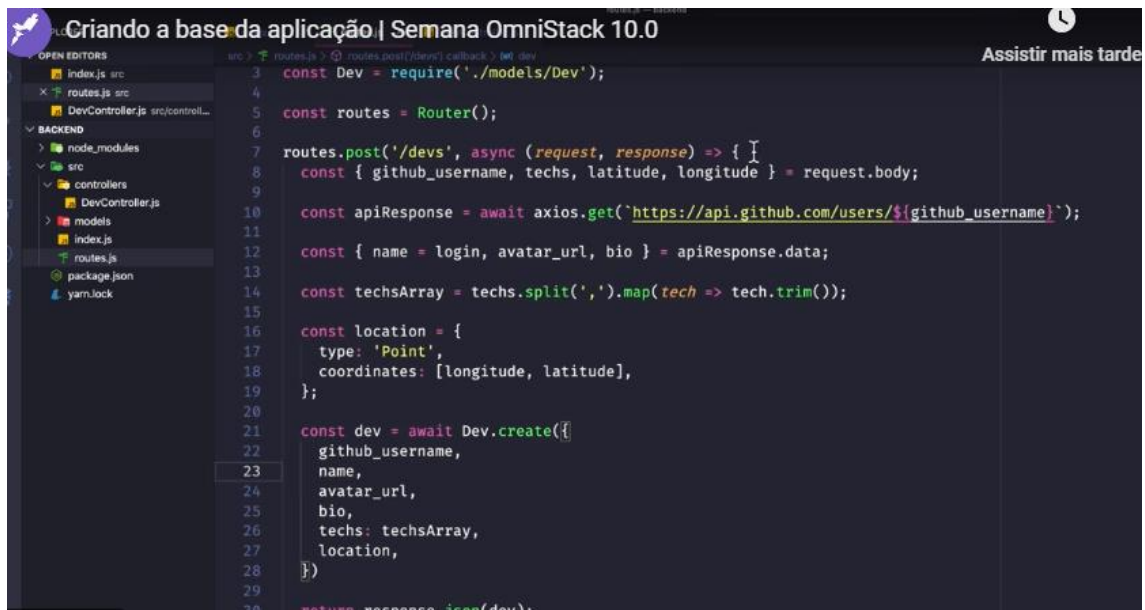
Tipo ponto no mapa

Refresh no mongodb p ver se funcionou



Pasta Camada controlers não é bom deixar rotas junto e criar arquivo devcontrollers.js

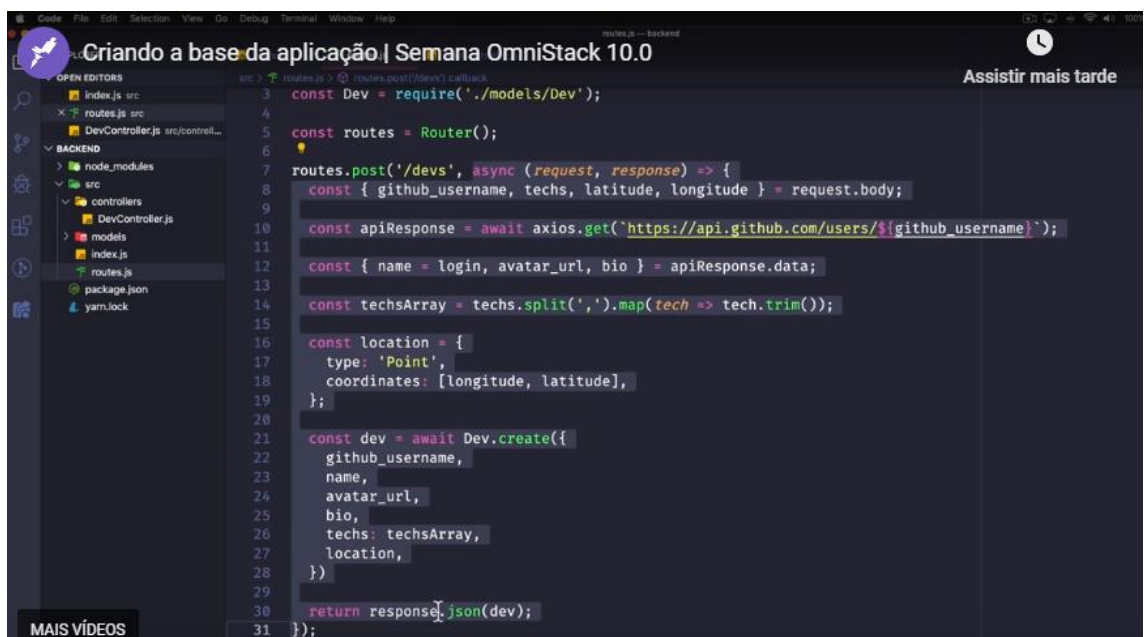




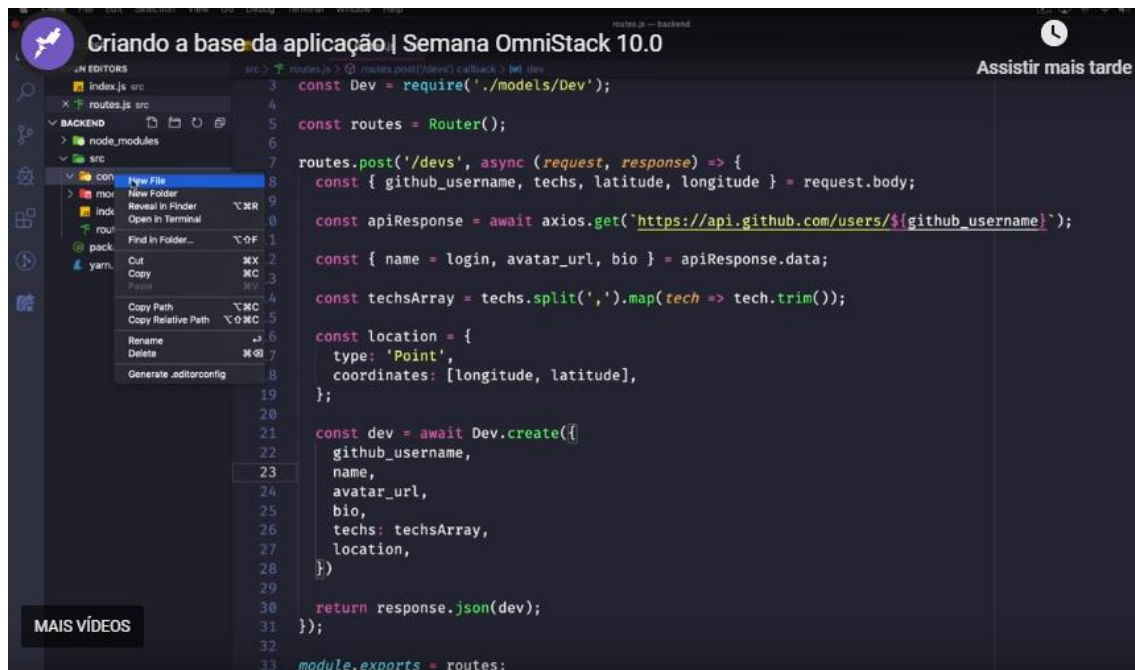
```
1 const Dev = require('./models/Dev');
2
3 const routes = Router();
4
5 routes.post('/devs', async (request, response) => {
6   const { github_username, techs, latitude, longitude } = request.body;
7
8   const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);
9
10  const { name = login, avatar_url, bio } = apiResponse.data;
11
12  const techsArray = techs.split(',').map(tech => tech.trim());
13
14  const location = {
15    type: 'Point',
16    coordinates: [longitude, latitude],
17  };
18
19  const dev = await Dev.create({
20    github_username,
21    name,
22    avatar_url,
23    bio,
24    techs: techsArray,
25    location,
26  });
27
28  return response.json(dev);
29
30 });
```

Controler serve para receber a requisição e devolver uma resposta

Mover esse código para devcontroller



```
1 const Dev = require('./models/Dev');
2
3 const routes = Router();
4
5 routes.post('/devs', async (request, response) => {
6   const { github_username, techs, latitude, longitude } = request.body;
7
8   const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);
9
10  const { name = login, avatar_url, bio } = apiResponse.data;
11
12  const techsArray = techs.split(',').map(tech => tech.trim());
13
14  const location = {
15    type: 'Point',
16    coordinates: [longitude, latitude],
17  };
18
19  const dev = await Dev.create({
20    github_username,
21    name,
22    avatar_url,
23    bio,
24    techs: techsArray,
25    location,
26  });
27
28  return response.json(dev);
29
30 });
```



Criando a base da aplicação | Semana OmniStack 10.0

```
const Dev = require('./models/Dev');

const routes = Router();

routes.post('/devs', async (request, response) => {
  const { github_username, techs, latitude, longitude } = request.body;

  const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);

  const { name = login, avatar_url, bio } = apiResponse.data;

  const techsArray = techs.split(',').map(tech => tech.trim());

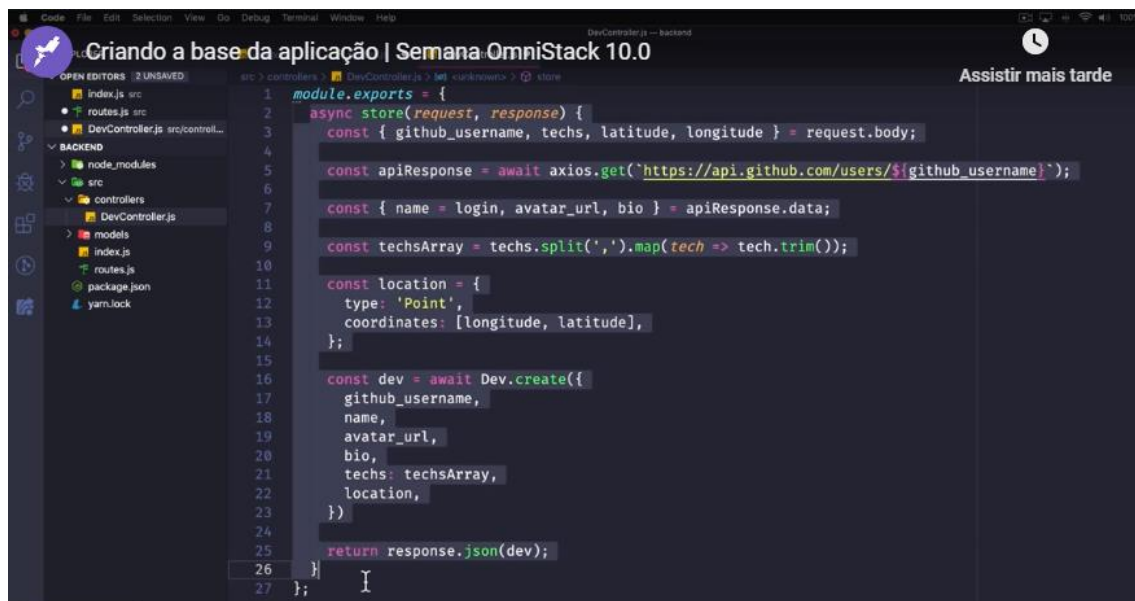
  const location = {
    type: 'Point',
    coordinates: [longitude, latitude],
  };

  const dev = await Dev.create({
    github_username,
    name,
    avatar_url,
    bio,
    techs: techsArray,
    location,
  });

  return response.json(dev);
});

module.exports = routes;
```

Nome para função async store (armazenar)



Criando a base da aplicação | Semana OmniStack 10.0

```
module.exports = {
  async store(request, response) {
    const { github_username, techs, latitude, longitude } = request.body;

    const apiResponse = await axios.get(`https://api.github.com/users/${github_username}`);

    const { name = login, avatar_url, bio } = apiResponse.data;

    const techsArray = techs.split(',').map(tech => tech.trim());

    const location = {
      type: 'Point',
      coordinates: [longitude, latitude],
    };

    const dev = await Dev.create({
      github_username,
      name,
      avatar_url,
      bio,
      techs: techsArray,
      location,
    });

    return response.json(dev);
  }
};
```